Spectral Graph-based Cyber Worm Detection Using Phantom Components and Strong Node Concept

Jamie L. Safar, Murali Tummala, and John C. McEachen
Department of Electrical and Computer Engineering
Naval Postgraduate School, Monterey, CA 93943
jamie.safar@nps.edu, mtummala@nps.edu, and mceachen@nps.edu

Abstract

Innovative solutions need to be developed to defend against the continued threat of computer worms. We propose the spectral graph theory worm detection model that utilizes traffic dispersion graphs, the strong node concept, and phantom components to create detection thresholds in the eigenspectrum of the dual basis. This detection method is employed in our proposed model to quickly and accurately detect worm attacks with different attack characteristics. It also intrinsically identifies infected nodes, potential victims, and estimates the worm scan rate. We test our model against the worm-free NPS2013 dataset, a modeled Blaster worm, and the WannaCry CTU-Malware-Capture-Botnet-284-1 and CTU-Malware-Capture-Botnet-285-1 datasets. Our results show that the spectral graph theory worm detection model has better performance rates compared to other models reviewed in literature.

1. Introduction

In the modern world, it has become increasingly difficult to secure and defend cyber systems. In 2019, the number of malware attacks reached 7.2 billion, the number of ransomware attacks reached 151.9 million [1], the number of phishing attacks that were successful in evading network security measures increased by 25 percent [2], and a 29 percent increase was observed in cryptocurrency attacks in the first quarter [3]. Additionally, adware aggressively increased to over 50 million detections across Windows and Mac systems. Yet, the number of consumer threat detections decreased by two percent [4].

Computer worms, specifically, present a significant threat to cyber systems due to the propensity of the attack to reach an alarming size in a very short time period [5, 6]. In May of 2017, the WannaCry ransomware cryptoworm infected over 200,000 hosts across 150 countries. In September 2018, it was estimated that more than 12,000 variants of the WannaCry worm were still being utilized [7]. Since attackers continue to evolve computer worms to be faster and more destructive, innovative solutions need to be developed that will shift the status quo towards, or in favor of, the network defender.

Several worm detection systems and methods have been developed, ranging from traditional anomaly detection methods, like local outlier factor (LOF) [8], to behavioral analysis methods, like destination-source correlation (DSC) [9]. However, the proposed systems and methods in literature leave room for improvement in detection rates, false alarm rates, and detection time. Additionally, most fail to provide additional critical information required for effective network response.

In this paper, we propose the novel spectral graph theory worm detection model (SGTWDM) that utilizes a traffic dispersion graph (TDG), the strong node concept (SNC), and phantom components to detect worm attacks. The SNC-based detection method employed in our proposed SGTWDM proves to be extremely accurate in detecting a worm attack once the second infected node begins exhibiting worm attack behavior. Additionally, the method intrinsically provides the network defender with the specific nodes that have been infected by the worm, other potential victims of the attack, and an estimated scan rate for the worm. These details provide the network defender with the required information to respond to an attack.

The remainder of this paper is organized as follows. Section 2 discusses related work on anomaly detection with specific focus on worm attacks. Section 3 proposes the SGTWDM and discusses the underlying components of that allow for detection of worm attacks. Section 4 shows the simulation results of the proposed detection method against a worm-free dataset, a modeled Blaster worm, and a WannaCry worm dataset. Finally, Section 5 provides a summary of the paper with discussion of future research.

URI: https://hdl.handle.net/10125/71468
978-0-9981331-4-0

2. Anomaly Detection Literature

In this section, we begin by providing an overview of the common detection algorithms utilized for intrusion detection. Then we discuss graph theory-based anomaly detection techniques proposed in literature review. Next, we examine other methods of worm attack detection in literature. Finally, we review the performance metrics of some of the worm detection methods proposed in literature.

2.1. Intrusion Detection

The detection of network security threats has been widely studied in the field of intrusion detection. Intrusion detection techniques can be categorized into one of two methods: signature-based (sometimes referred to as misuse-based) or anomaly-based. Signature-based methods require known profiles (or signatures) of the attack which are utilized as references to detect future attacks. These methods are very accurate, having a low false-alarm rate, but are not effective against previously unknown attacks.

Anomaly-based methods on the other hand, utilize statistical behavior modeling techniques to determine normal from abnormal [10]. These methods are typically prone to higher false-alarm rates, unless the network follows strict/static behavioral patterns [10, 11]. They also detect previously unknown attacks. Anomaly-based methods can be divided into three categories according to the methods utilized for detection: statistical, knowledge (or data mining), and machine learning [10]. A classification of the different algorithms of each of these categories found in literature is shown in Figure 1.

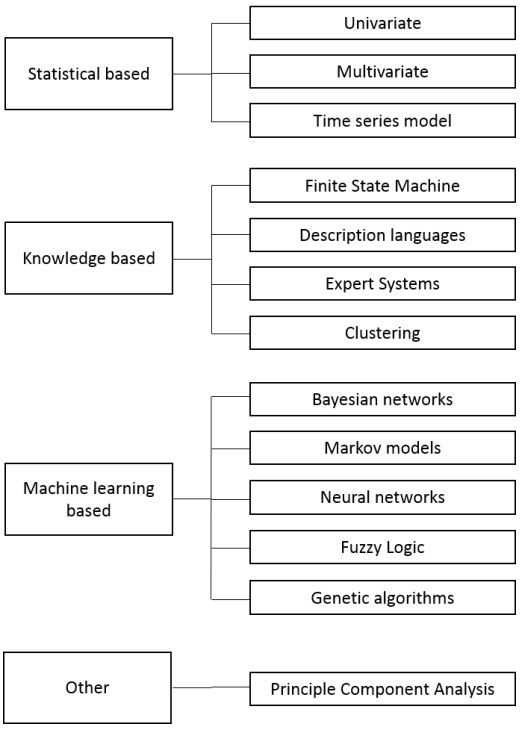

Figure 1. Classification of anomaly based IDSs according to their detection algorithm, after [10, 12].

2.2. Graph Theory-based Detection

Since our worm detection method utilizes graph theory and spectral graph theory tools, we reviewed literature for other graph theory-based anomaly detection methods and discovered three proposed methods [20, 22, 23]. The work in [22] uses TDGs to model network traffic and to train the proposed detection system. The system training produces graph edit distance and dK-2 distance thresholds that are used for detection. Flows that are detected as anomalous are passed to the classification system, which utilizes the VF2 graph matching algorithm, a graph-subgraph isomorphism analysis algorithm, to compare the graph of the anomalous flows to known graphical attack patterns. While this method is successful, it assumes all graphical attack patterns are known without explanation of how those are acquired. Additionally, this system requires increased computational overhead in detecting and classifying a worm attack, since two separate algorithms are performed.

The work in [20] uses the causal-tree worm propagation behavior to detect worm attacks. Specifically, the detection system uses various graph metrics, like branching factor and depth, to identify worm causal-tree behavior. While the method presented in [20] is successful in identifying an attack, it fails to inherently identify the specific nodes infected by the attack.

Finally, the work in [23] uses spectral graph tools to detect network congestion in a software-defined network's physical layer. The detection methods employs phantom nodes to create a threshold in the eigenvector matrix, which is utilized to compare node connectivity, or available link capacity, to the connectivity of the phantom node. While the work provides a method to detect network congestion, the method can not distinguish normal network congestion from congestion resulting from an attack.

Our proposed SGTWDM adapts and expands upon these graph theory-based works. The result is a more computationally efficient worm detection system that inherently identifies the infected node(s), the potential victim nodes, and an estimated attack scan rate, for the

Table 1. Additional worm detection methods in literature.

Source	Target	Detection Method
[9]	Scan-based, fast spreading, local worm	Two-phased DSC algorithm
[8]	Worms, DoS, probe	LOF association mining
[13]	Worms, probe, DoS, alpha flows	Multiway-subspace methods using feature entropy
[14]	Internet worm	Time of next infection
[15, 16]	Camouflaging worms	Power spectral density distribution and spectral flatness measure
[17]	Worms, distributed denial of service, and botnets	Collaborative intrusion prevention architecture (CIPA) with neural net
[18]	Worms, distributed denial of service, etc.	Connection attempt and host unreachable message count

Table 2. Worm detection performance metrics in literature.

Source	Detection Rate	False Alarm Rate	Detection Time
[19]	Before 1.5% of vulnerable nodes infected	N/A	N/A
[9]	When 0.0064 of vulnerable nodes infected	N/A	N/A
[13]	0.8	N/A	N/A
[14]	N/A	N/A	100 min
[20, 21]	N/A	0.015 (theoretical)	1.8 sec (theoretical)
[15, 16]	0.993	N/A	1460 min
[17]	Witty: 0.96 Slammer: 0.95 Conficker: 0.92	Witty: 0.05 Slammer: 0.075 Conficker: 0.095	N/A

purpose of network defense.

2.3. Other Worm Detection Methods

For comprehensive purposes, we reviewed other methods of detection for worm attacks. Statistical methods based on connection failures, probes of unused address space, and in/out packet counts are commonly utilized due to the underlying scanning behavior of the worm. Berk et al. [24] proposed an internet control message protocol (ICMP) destination unreachable based detection algorithm, and the distributed anti-worm (DAW) architecture [25] utilized a detection algorithm that tracks transmission control protocol TCP synchronize (SYN) and reset (RST) packets for the detection of worm attacks. However, both are limited by the type of worm it will detect, require a large amount of resources, and are unable to detect the worm until a significant infection has already been achieved [26]. Wu et al. [19] proposed a victim-based algorithm that classifies a source as a victim if it sends packets to inactive addresses and then sets a threshold on the number of victims in order to generate an alert for the presence of a worm. While this method requires less resources, it is still not very useful for detecting other types of worms or network attacks. Additional worm detection methods not mentioned in this section can be found in Table 1.

2.4. Worm Detection Metrics

With respect to performance metrics, very few worm detection systems and methods in literature provide analytical analysis on detection rate, false alarm rate, and detection time. Table 2 provides the worm detection metrics we observed in literature. For most methods, the detection rates are above 0.9 and the false alarm rates are below 0.1; however, very few methods provide information on the time to detect the attack.

3. Proposed Spectral Graph Theory Worm Detection Model

In this section, we present the SGTWDM shown in Figure 2. We first explore the critical components utilized in the SNC-based detection method. Specifically, we discuss TDGs as a means of representing network traffic, phantom components as a method to produce detection thresholds in the eigenspectrum index of the eigenvalue matrix Λ and eigenvector matrix V, and the SNC as a method to characterize network traffic behavior. Then, we describe the SNC-based detection process. Next, we discuss parameters determined externally by the network administrator. Finally, we present the attack parameters identified through the detection process. We limit our discussion in this section to scan-based TCP worms, but the methods and model are also applicable for user datagram protocol (UDP) worms.

3.1. Traffic Dispersion Graph

Adopting the TDG method in [22] to mathematically represent network traffic, the proposed SGTWDM utilizes TDGs to mathematically and visually represent end-to-end communications within the network. In a

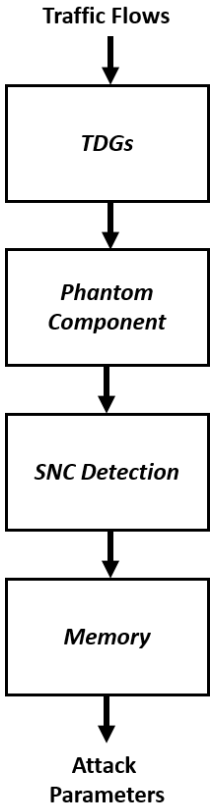

Figure 2. Functional components of the SGTWDM.

TDG, the source and destination internet protocol (IP) addresses are defined as nodes, and the network flow (or communication) between the source and destination is defined as a link. The TCP tuple is used to identify the source and destination nodes of a network flow [22].

Network logical flows are used to create the adjacency matrix A, degree matrix D, and Laplacian matrix Q. The $n \times n$ adjacency matrix is defined as

$$A(i,j) = \begin{cases} w_{i,j} & \text{if } a_{i,j} \in L, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where $w_{i,j}$ is the link weight between the i^{th} and j^{th} nodes, $a_{i,j}$ is the i^{th} element of the j^{th} column of A, and L is the set of all links. In the case of the SGTWDM, $w_{i,j} = 1$ if a logical flow exists between a pair of nodes, and $w_{i,j} = 0$ if a logical flow does not exist between a pair of nodes. The $n \times n$ degree matrix is defined as

$$D(i,j) = \begin{cases} \sum_{j=1}^{n} w_{i,j} & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

The $n \times n$ Laplacian matrix is defined as

$$Q = D - A, \quad (3)$$

or [27]

$$Q(i,j) = \begin{cases} -w_{i,j} & \text{if } a_{i,j} \in L, \\ \sum_{j=1}^{n} w_{i,j} & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

The Laplacian matrix contains all necessary information to visually reconstruct the network data into graphical format.

It is important to note, not all network traffic flows are utilized by the model to form TDGs. To conform with the worm attack behavior (see Section 3.2), the traffic flows are filtered to keep data on SYN-only packets, the first packet the of the TCP 3-way handshake, with unique destination addresses. As an added benefit, the eigendecomposition computational costs associated with the SNC-based detection method are greatly reduced by this filtering process.

3.2. Phantom Component

Expanding upon techniques utilized in [23], we developed the phantom component. The phantom component is a mathematical construct of a cluster of connected nodes that does not physically exist in the traffic flows of the TDG, but exists within A. The phantom component is designed to provide a graphical representation of a known attack behavior.

To develop the phantom component for worm attacks, we draw upon the causal tree behavior presented in [20]. Figure 3 shows the graphical behavior of a worm attack using the behavior of an infected node during the target search and discovery process. During this process, the infected node attempts to discover a new target by sending a SYN packet as part of the TCP three-way handshake. The proposed phantom component for a worm attack only utilizes the first level of offspring from the graphical behavior of the worm in Figure 3. Additionally, the flow direction of the links in the graphical behavior is disregarded since the phantom component resides in the undirected, symmetric A. In order to achieve additional levels of offspring from the phantom component, memory must be introduced to store detected worm attack behavior (see Section 3.5).

It is important to note, we considered a phantom component with more levels of offspring and no system memory, but found through initial analysis that this generally increased the false alarm rates. Additional research needs to be conducted to determine the cause of the increase.

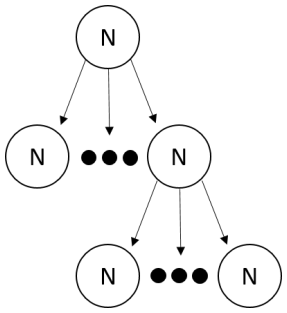

Figure 3. Graphical representation of the behavior of a worm attack.

3.3. Strong Node Concept

The SNC uses spectral graph theory tools to associate a node's connectivity to the element values within V. While this concept is utilized in [23], the SNC has never been formally defined, and the mathematical foundation for the SNC has not previously been provided.

Spectral graph theory uses Q to solve for the eigenvalues and eigenvectors, which provide spectral analysis of the graph characteristics. The eigenvalues and eigenvectors are defined as the solution to

$$Qv_i = \lambda_i v_i, \quad i = 1, 2, ..., n \tag{5}$$

where λ_i is the i^{th} eigenvalue, v_i is the corresponding $n \times 1$ eigenvector, i is the index of the eigenspectrum, and the order of the eigenvalues is [27]

$$0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_{n-1} \leq \lambda_n. \tag{6}$$

Utilizing these spectral graph theory tools, we have developed a mathematical foundation for the SNC. Given (3), (5) can be rewritten as

$$0 = d_i v_{i,j} - (\sum_{k=1}^{n} a_{i,k} v_{k,j} \mid k \neq i) - \lambda_j v_{i,j} \tag{7}$$

or

$$v_{i,j} = \frac{1}{d_i - \lambda_j} (\sum_{k=1}^{n} a_{i,k} v_{k,j} \mid k \neq i) \tag{8}$$

where d_i is the degree associated with the i^{th} node, and v_{ij} is the i^{th} element in the j^{th} vector of V [28].

Holding all other variables constant, d_i and λ_j provide insight into the behavior of $v_{i,j}$. Given a network containing six nodes where $d_1 \leq d_2 \leq d_3 \leq d_4 \leq d_5 \leq d_6$, the magnitude of the scaling factor $\frac{1}{d_i - \lambda_j}$ in (8) will be larger for the smallest degree nodes in the eigenvector associated to λ_2 compared to the largest degree node. Conversely, the magnitude of the scaling factor $\frac{1}{d_i - \lambda_j}$ will be larger for the largest degree nodes in the eigenvector associated to λ_6 compared to the smaller degree node. As depicted in Figure 4, this results in the smallest degree node having a larger magnitude in $v_{i,2}$ and the largest degree node having a larger magnitude in $v_{i,6}$. We expand this line of reasoning in Figure 5 to show how node connectivity directly impacts the magnitude of the value in each $n \times 1$ vectors of V.

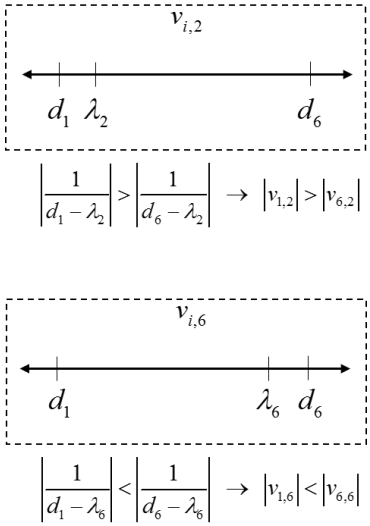

Figure 4. Visual number line depiction of effects of scaling factor $\frac{1}{d_i - \lambda_j}$ from (8) on the magnitude values in v_{ij}.

Using deductive reasoning to expand upon these results, the summation $(\sum_{k=1}^{n} a_{ik} v_{kj} \mid k \neq i)$, from (8), is directly influenced by the degree of the node and the connectivity of its neighbors. In other words, a node with a higher degree will have more terms in the summation than a smaller degree node. However, the terms are directly scaled by the neighboring node's value in the eigenvector being considered, which in turn is being scaled by $\frac{1}{d_i - \lambda_j}$. As a result, the scaling factor $\frac{1}{d_i - \lambda_j}$ in (8) appears to be the dominating factor of the equation. Consequently, the effects depicted in Figure 5 should hold true across all networks and graphs.

3.4. SNC-based Detection

The SGTWDM combines the phantom component with the SNC to detect worm attacks. In general, a phantom component threshold $T_{phantom}$ (see Section 3.6) is utilized to establish a threshold $T_{eigenindex}$

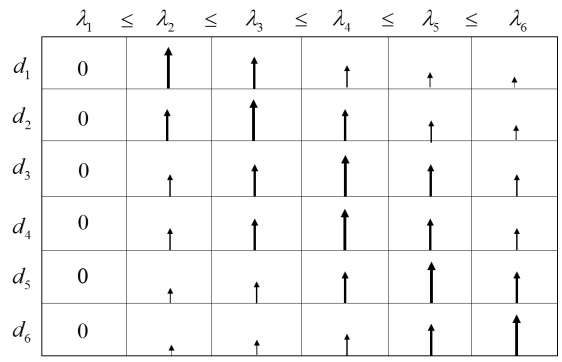

Figure 5. Effects of scaling factor $\frac{1}{d_i - \lambda_j}$ from (8) on the magnitude of the values in v_{ij} given $d_1 \leq d_2 \leq d_3 \leq d_4 \leq d_5 \leq d_6$.

within the eigenspectrum of Λ and V. Specifically, the phantom component will have the primary nodal influence $|v_{i,j}^{primary}|$, defined as the largest $|v_{i,j}|$ in eigenspectrum index j, in one or more of the $n \times 1$ column-vectors of V. The eigenvalue $\lambda_{phantom}$ associated with the largest eigenspectrum index where the phantom component has $|v_{i,j}^{primary}|$ is then utilized to establish $T_{eigenindex}$. Specifically, $T_{eigenindex}$ is equal to the smallest eigenspectrum index that has an eigenvalue equal to $\lambda_{phantom}$. As a result, nodes that have $|v_{i,j}^{primary}|$ in an eigenspectrum index greater than or equal to $T_{eigenindex}$ are detected as infected and exhibiting worm attack behavior. Additionally, all other non-zero $|v_{i,j}|$ in that eigenspectrum index are identified as potential victims of the worm attack.

3.5. Memory

Since only a single level of offspring is utilized in the phantom component, memory must be introduced into the detection method to detect a second level of offspring. This in turn reduces the number of the false alarms by distinguishing worm attack behavior from normal network traffic.

The offspring, or potential victims, of an identified infected node are stored in memory for a specified length of time t_{log} (see Section 3.6). If one of the stored offspring nodes is detected as an infected node in a subsequent detection window, then the SGTWDM detects a worm attack and outputs the attack parameters (see Section 3.7). As a result, the SGTWDM is designed to detect worm attacks once the second node is infected. This means the first node infected by the worm will not be identified as infected until a second node is infected and begins scanning for new victims.

3.6. External Parameters

There are three parameters critical to the SGTWDM that must be externally determined by the network administrator. The first two parameters are the window length L_w and $T_{phantom}$. L_w controls the amount of traffic flows that are processed through the system and should be as small as possible to reduce the computational costs and provide quick identification of worm attacks. $T_{phantom}$ represents the number of offspring in the phantom component and should be as large as possible to keep benign network traffic below the SNC detection threshold. The ratio give by

$$\frac{T}{L_w} < R_{scan} \tag{9}$$

must also hold true, where R_{scan} is the worm scan rate.

The third parameter is length of time t_{log} the first-level offspring (and its parent) are logged and stored in memory. In general, the relationship given by

$$t_{log} > t_T - t_{SYN} \tag{10}$$

must be valid, where t_T is the time when the newly infected victim sends $T_{phantom}$ or more SYN packets in the defined L_w, and t_{SYN} is the time when the newly infected node first received the SYN-only packet of the TCP three-way handshake from its infector.

3.7. Attack Parameter Outputs

The SGTWDM not only detects a worm attack among normal network traffic, it also intrinsically provides the exact nodes that are infected (the parent nodes from the graphical flows that produced a worm alarm), potential additional victims (the offspring nodes from the graphical flows that produced a worm alarm), and an estimated scan rate of the worm given by

$$R_{scan_{est}} = \frac{O_2}{L_w}, \tag{11}$$

where O_2 is the number of second-level offspring of the graphical flows that produced a worm alarm. These three attack parameters provide network defenders with critical information to defend against the worm attack. It even provides the opportunity to externally pre-program network responses to attain quicker response times.

4. Results

In this section, we provide the results from three sets of simulations using MATLAB. The first and second simulations examine the SGTWDM detection

Table 3. Performance metrics of the SGTWDM against the Blaster worm.

L_w	Detection rate	Detection time (from 2^{nd} node infection)	Scans sent by 2^{nd} node prior to detection	Size of infection at time of detection
1	1.00	0.42 seconds	20	2
2	1.00	0.92 seconds	20	2
3	1.00	0.92 seconds	20	2
4	1.00	0.92 seconds	20	2

Table 4. Performance metrics of the SGTWDM against the WannaCry worm.

L_w	Detection rate	Detection time (from 2^{nd} node infection)	Scans sent by 2^{nd} node prior to detection	Size of infection at time of detection
1	1.00	6.16 seconds	15	2
2	1.00	4.16 seconds	9	2
3	1.00	2.16 seconds	5	2
4	1.00	2.16 seconds	5	2

rate and detection time for datasets containing a Blaster worm and a WannaCry worm, respectively. For these simulations, we focus on the windows around the time that the second node is infected since the SGTWDM cannot detect an attack until the second node is infected and begins exhibiting worm attack behavior. We select different L_w that align with (9) and are small enough to provide quick identification of a potentially fast spreading attack. The third simulation examines a larger network traffic dataset containing no worm attack to determine the system's ability to distinguish normal network traffic from attack traffic. For all simulations, we set $t_{log} = 40$ seconds to account for worms that require a system reboot as part of the infection process [29, 30], and $T_{phantom} = 5$ to align with (9) and evaluated characteristics of numerous historical worm attacks.

4.1. Blaster Worm Model

For the first simulation, we developed a model of the Blaster worm utilizing specifications from [29, 30]. We used MATLAB to produce Blaster worm traffic, and then spliced the worm attack traffic into 60 minutes of normal network traffic flows collected from the Naval Postgraduate School (NPS) network in 2013. For the remainder of this paper, this collection of NPS campus flows is referred to as the NPS2013 dataset. The NPS2013 dataset contains 1,314 nodes and has an estimated average SYN transmission rate of 16.17 packets per second. The final infection size of the Blaster worm is 11 nodes at the end of 60 minutes.

For our results, we focus on minute 38 to minute 39 of the data where the second node is infected by the Blaster worm. The performance metrics of the SGTWDM against the Blaster worm are shown in Table 3. For all evaluated L_w, the detection rate was one and the time to detect the second infected node after it began exhibiting worm attack behavior was less than one second. These results improve upon the detection rates from our literature review (presented in Table 2). It is important to note, our detection time is from the second node infected, whereas the detection times in Table 2 are from the initial infection. As a result, we are unable to fairly compare our detection times to those in Table 2.

Since the worm attack was detected when the infection size was only two nodes, a response mechanism could be implemented by the network administrator to potentially stop the worm attack before it reached it's final size of 11 nodes. It is important to note, the SGTWDM also accurately and quickly detected each of the nine subsequently infected nodes once they began exhibity worm attack behavior.

4.2. WannaCry Worm Dataset

For the second simulation, we utilized the WannaCry datasets: CTU-Malware-Capture-Botnet-284-1 [31] and CTU-Malware-Capture-Botnet-285-1 [32]. Combining the two datasets, we focused on the data between 450 and 515 seconds where the second node is infected by the Wannacry worm. The final infection size of this dataset is 2 nodes. The performance metrics of the SGTWDM against the WannaCry worm are shown in Table 4. For all evaluated L_w, the detection rate was one but the times to detect the infection ranged from just over two second to just over six seconds. The detection results improve upon the detection rates in from our literature review, presented in Table 2, but at the expense of a longer detection time. Again, it is important to note, our detection time is from the second node infected, whereas the detection times in Table 2 are from the initial infection. As a result, we are unable to fairly compare our detection times to those in Table 2.

Compared to the Blaster worm simulation, the time to detect the infection was greater due to the initially slow scan rate of the second node after it's infection. This indicates that the SGTWDM performs better with a smaller L_w against worms that have a faster ramp up in scan behavior compared to worms that have a slower ramp up in scan behavior. Worms that have a slower ramp up in scan behavior require a longer L_w to decrease the detection time.

4.3. System False Alarm Rate

For the final simulation, we ran the NPS2013 dataset through our SGTWDM to determine the system false alarm rates. We evaluated the false alarm rates for $L_w = \{0.5, 1, 2, 3, 4, 5\}$ seconds. These specific L_w were selected based on two criteria: 1) alignment with the L_w utilized in the first two simulations, and 2) requirement to provide quick identification of a potentially fast spreading attack. The false alarm rate results of this simulation are shown in Figure 6. As expected, the false alarm rate increases as L_w increases; a longer L_w provides more opportunity for normal traffic to send a number of SYN packets greater than or equal to $T_{phantom}$. As a result, a smaller L_w needs to be implemented by the network administrator to keep false alarm rates low. Still, even at $L_w = 5$, the false alarm rate is only 0.001 greater than the theoretical rate in [20, 21] and vastly improves upon [17] (see Table 2).

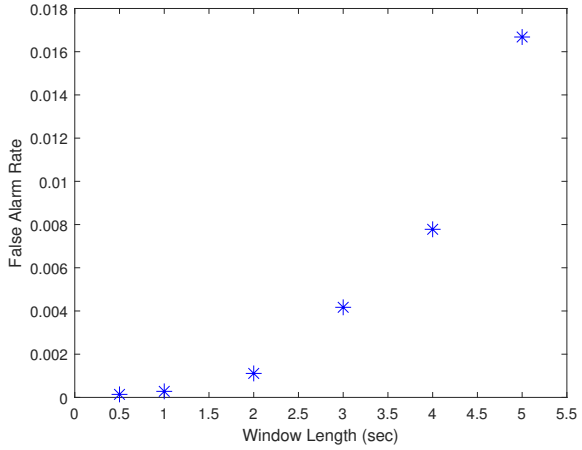

Figure 6. False alarm rates of the SGTWDM using normal traffic flows from the NPS2013 dataset.

5. Conclusions and Future Work

This paper proposed a novel SGTWDM that utilizes spectral graph theory tools to quickly and accurately detect worm attacks. We discussed the underlying components of the model, including TDGs, phantom components, and the SNC. We provided a mathematical foundation to support the SNC, and described the SNC-based detection method employed in the SGTWDM. We then simulated the model in MATLAB and evaluated the model using the NPS2013 dataset, a modeled Blaster worm, and a WannaCry worm dataset to determine the detection rates, false alarm rates, and times to detection. Using the Blaster worm and WannaCry datasets, the SGTWDM produced a detection rate of one once the second infected node began exhibiting worm attack behavior. From the NPS2013 dataset, we determined that the false alarm rate of the SGTWDM increases as L_w increases. Regardless, the results improve upon the other detection methods examined in our literature review.

Our future work will focus on developing a system that utilizes the SGTWDM to detect and classify both worm and distributed denial of service attacks. The system will be theoretically analyzed to develop upper and lower bounds on the system's false alarm rates. Additionally, research needs to be conducted on the effects of perturbation on the network traffic to the spectral graph-based detection method.

References

[1] SecurityMagazine. (2019, Oct.) First three quarters of 2019: 7.2 billion malware attacks, 151.9 million ransomware attacks. Security Magazine. [Online]. Available: https://www.securitymagazine.com/articles/91133

[2] J. Davis. (2019, Jul.) Phishing attacks on the rise, 25% increase in threats evading security. Health IT Security. [Online]. Available: https://healthitsecurity.com/news/phishing-attacks-on-the-rise-25-increase-in-threats-evading-security

[3] M. Boddy. (2019, Aug.) Report: Cryptojacking campaigns up by 29%, ransomware attacks up 118%. COINTELEGRAPH. [Online]. Available: https://cointelegraph.com/news/report-cryptojacking-campaigns-up-by-29-ransomware-attacks-up-118

[4] MalwarebytesLabs. (2020, Feb.) 2020 state of malware report. MalwareBytes Labs. [Online]. Available: https://resources.malwarebytes.com/files/2020/02/2020_State-of-Malware-Report.pdf

[5] F. Wang, Y. Zhang, C. Wang, J. Ma, and S. Moon, "Stability analysis of a SEIQV epidemic model for rapid spreading worms," *Computers Security*, vol. 29, pp. 410–418, 2010.

[6] J. L. Safar, M. Tummala, J. C. McEachen, and C. Bollmann, "Modeling worm propagation and insider threat in air-gapped network using modified seiqv model," in *2019 13th International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2019, pp. 1–6.

[7] NakedSecurity. (2018, Sep.) Wannacry the worm that just wont die. SOPHOS. [Online]. Available: https://nakedsecurity.sophos.com/2019/09/18/wannacry-the-worm-that-just-wont-die/

[8] L. Ertoz, E. Eilertson, A. Lazarevic, P.-n. Tan, V. Kumar, and J. Srivastava, "Minds minnesota intrusion detection system," in *NextGeneration Data Mining*, H. Kargupta, J. Han, P. S. Yu, R. Motwani, and V. Kumar, Eds. Cambridge, MA: MIT Press, 2004, ch. 3, pp. 199–218.

[9] G. Gu, M. Sharif, X. Qin, D. Dagon, W. Lee, and G. Riley, "Worm detection, early warning and response based on local victim information," in *20th Annual Computer Security Applications Conference*, 2004, pp. 136–145.

[10] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[11] O. Kachirski and R. Guha, "Effective intrusion detection using multiple sensors in wireless ad hoc netowrks," in *Proceedings of 36th Annual Hawaii International Conference on System Sciences (HICSS)*, 2003, p. 57.

[12] G. á Teodoro, J. D. áz Verdejo, G. M. Fernańdez, and E. Vaźquez, "Anomaly-based network intrusion detection: techniques, systems and challenges," *Computers and Security*, vol. 28, pp. 18–28, 2009.

[13] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *Computer Communication Review (CCR)*, vol. 35, 10 2005, pp. 217–228.

[14] D. Nicol, "The impact of stochastic variance on worm propagation and detection," in *WORM '06: Proceedings of the 4th ACM workshop on Recurring malcode*, New York, NY, 2006, pp. 57–64.

[15] W. Yu, X. Wang, P. Calyam, D. Xuan, and W. Zhao, "On detecting camouflaging worm," in *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*, 2006, pp. 235–244.

[16] ——, "Modeling and detection of camouflaging worm," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 3, pp. 377–390, 2011.

[17] X.-F. Chen and S.-Z. Yu, "Cipa: A collaborative intrusion prevention architecture for programmable network and sdn," *Computers Security*, vol. 58, pp. 1–19, 05 2016.

[18] A. A. Cardenas, J. S. Baras, and V. Ramezani, "Distributed change detection for worms, ddos and other network attacks," in *Proceedings of the 2004 American Control Conference*, 2004, pp. 1008–1013.

[19] J. Wu, S. Vangala, L. Gao, and K. Kwiat, "An effective architecture and algorithm for detecting worms with various scan," in *Proceedings of the 11-th IEEE Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, Feb 2004.

[20] D. R. Ellis, J. G. Aiken, A. M. McLeod, and D. R. Keppler, "Graph-based worm detection on operational enterprise networks," MITRE, Tech. Rep. MTR-06W0000035, Apr. 2006.

[21] D. Ellis, J. Aiken, K. Attwood, and S. Tenaglia, "A behavioral approach to worm detection," in *Proceedings of the 2004 ACM Workshop on Rapid Malcode, WORM 2004*, Washington, DC, USA,, October 2004, pp. 43–53.

[22] D. Q. Le, T. Jeong, H. E. Roman, and J. W. Hong, "Traffic dispersion graph based anomaly detection," in *Proceedings of the Second Symposium on Information and Communication Technology*, 2011, pp. 36–41.

[23] J. Johnson, "Software defined network monitoring scheme using spectral graph theory and phantom nodes," M.S. thesis, Dept. of Electrical and Computer Engineering, Naval Postgraduate School, Monterey, CA, 2014.

[24] G. B. V. Berk and R. Morris, "Designing and framework for active worm detection on global networks," in *Proceedings of 1st IEEE International Workshop on Information Assurance*, 2003.

[25] S. C. Y. Tang, "Slowing down internet worms," in *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, Washington, DC, USA, 2004, pp. 312–319.

[26] P. Li, M. Salour, and X. Su, "A survey of internet worm detection and containment," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 1, pp. 20–35, 2008.

[27] P. V. Mieghem, *Graph spectra for complex networks*. Cambridge, NY: Cambridge University Press, 2011.

[28] K. C. Das, "The laplacian spectrum of a graph," *Computers and Mathematics with Applications*, vol. 48, pp. 715–724, 2004.

[29] J. V. Hoogstraten. (2003) Malware faq: What is w32/blaster worm? SANS. [Online]. Available: https://www.sans.org/security-resources/malwarefaq/w32-blasterworm

[30] K. Kasten. (2018, Mar.) What how long does a reboot take? Quora. [Online]. Available: https://www.quora.com/What-how-long-does-a-reboot-take

[31] StratosphereLab. (2017, Jul.) Index of /publicdatasets/ctu-malware -capture-botnet-284-1. Virus Total. [Online]. Available: https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-284-1/

[32] ——. (2017, Jul.) Index of /publicdatasets/ctu-malware -capture-botnet-285-1. Virus Total. [Online]. Available: https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-285-1/